LA-UR-

99-223

Approved for public release;
distribution is unlimited.

CONF-980412--

Title: Adapting Perspectives to Facilitate Knowledge Assimilation

Author(s): Kelsey, Robert L.-XCM

Webster, Robert B.-XCI

Submitted to: SPIE Aerosense
Orlando, FL
April 1998

RECEIVED

MAR 12 1999

OSTI

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

MASTER

Los Alamos

NATIONAL LABORATORY

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the U.S. Department of Energy under contract W-7405-ENG-36. By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Adapting perspectives to facilitate knowledge assimilation

R. L. Kelsey^a, R. B. Webster^b

Los Alamos National Laboratory,

^aXCM MS-F645, Los Alamos, NM 87545

^bXCI MS-F663, Los Alamos, NM 87545

ABSTRACT

The notion of perspective when supported in knowledge representation can allow the representation of multiple and varying points of view, some of which may even be inconsistent with one another. In an object-based knowledge representation methodology created and used by the authors, a perspective is defined by consolidating a number of objects and a number of those objects' associated attributes and methods into a view. This view can help partition a knowledge domain into separate portions. A separate portion represents an individual's view of the knowledge domain. Representation of multiple and varying perspectives can add detail and context to the knowledge in a knowledge domain. The ability to create new perspectives may add to the existing knowledge as well as reveal paths to additional knowledge.

A simple example is presented where perspectives are used to represent game playing strategies and levels of expertise in those strategies. Players' perspectives are adapted and changed to provide additional knowledge and insight into further game playing strategies. Results show improvement in the playing of the games. Additionally, a more complex problem for applying these techniques is introduced.

Keywords: knowledge assimilation, abstraction, perspective, object-based knowledge representation

1. INTRODUCTION

Correct problem representation can make reasoning about and solving a problem easier.¹ Representation in this case can be thought of as an abstraction, point of view, or perspective. Anderson gives several examples of how thinking about a problem differently or thinking about state and operators in the problem differently can make the problem easier to solve.¹ An example is the two-string problem of Maier.² There are many versions of this problem, but essentially there are two strings hanging apart from each other from the ceiling of a room. Also in the room is a chair and a pair of pliers. The strings are far enough apart and hang high enough that a human cannot grab both (one in each hand) at the same time. The objective is for the human to tie the hanging ends of the two strings together using only what is available in the room.

The solution to the two-string problem is to tie the pliers to one of the strings and set it swinging like a pendulum. Then get the other string and carry it towards the swinging string, stand on the chair and grab the swinging string, and tie the ends together. This is a tricky problem to solve unless the representation of the items in the room is abstracted, specifically, the representation of the use of the pliers. To solve the problem, the pliers are used as a weight to help swing one of the strings. This is not the normal representative use of a pair of pliers, but it is the correct one for this problem.¹

When a number of abstractions or perspectives are available, they can be shared and used as a learning tool. Presenting additional perspectives to a problem can lead a person to further detail and insight, which may lead to increased understanding of the problem. If the presentation of such perspectives is accomplished in an intelligent manner, then they can be used as a training guide or procedure.

The following sections discuss a demonstration and experiment of guiding users to further understanding of a problem by adapting their initial perspective. The example problem is a simple one and was chosen because of how well it demonstrates the importance and potential benefits of recognizing and learning a new perspective on a problem.

(Send correspondence to R.L.K.)

R.L.K.: E-mail: rob@lanl.gov

R.B.W.: E-mail: robw@lanl.gov

There are eight possible ways to win at the word game which comprise the eight combinations of the nine words sharing a letter. These nine words can also be arranged in the nine squares of a three-by-three matrix and then played like tick-tack-toe. Figure 3 shows the numbers of the fifteen game arranged in a three-by-three magic square and the words of the word game arranged in a three-by-three matrix.

$1 + 5 + 9 = 15$	$2 + 4 + 9 = 15$	$2 + 6 + 7 = 15$	$3 + 5 + 7 = 15$
$1 + 6 + 8 = 15$	$2 + 5 + 8 = 15$	$3 + 4 + 8 = 15$	$4 + 5 + 6 = 15$

Figure 2. The eight combinations of three numbers adding to 15.

These abstractions to tick-tack-toe can be represented as perspectives. When a player's perspective is changed or adapted to the tick-tack-toe abstraction, then the fifteen game and word game should be easier to play. The player can exercise the strategy to play tick-tack-toe without the extra concentration needed for addition and/or subtraction for the fifteen game and consideration of letters for the word game.

8	1	6	HOT	FORM	WOES
3	5	7	TANK	HEAR	WASP
4	9	2	TIED	BRIM	SHIP

Figure 3. The numbers of the fifteen game and the words of the word game in tick-tack-toe format.

2.3. An experiment

An experiment was designed where a player plays the three different games against a computer. The experiment begins with a player playing a series of tick-tack-toe games. The number of games in the series is determined by how well the player plays. If the player wins, no more games are necessary. At least two games are played in the event of a tie and a loss means one more game is played. To curtail impatience, no more than three games are played in a series.

The second series of games is the fifteen game and the third series is the word game. The player may decide to make the first move in each game or allow the computer to make the first move. If the player experiences problems playing the fifteen game and word game (respectively) as evidenced by loss/tie statistics then an additional series of games is presented. In the additional series, hints are provided to abstract the fifteen game and word game into the game of tick-tack-toe. The number of games in these series is again based on how well the player plays. If the player wins or ties, the series will end.

The three games, their presentation to a player, and the consolidating experimental process are all implemented using a Web-based implementation. A volunteer player participates by using a Web browser to view brief instructions and play the games. An identification name for the player is recorded along with the player's win/loss/tie scores and the number of moves for each. These statistics are used to match a player with a particular perspective and to determine if and when to adapt that perspective.

2.4. Experimental Results

In general, most players experienced tie games while playing tick-tack-toe. Some players were even able to beat the computer due to a flaw in the algorithm. It was decided to not correct the flaw so that this extra level of expertise could be recorded. When a player won, the next type of game was immediately presented. Had a player won at tick-tack-toe, the fifteen game, and the word game, the player would not have been asked to play the abstracted versions of the fifteen game and the word game. These type of win statistics would have suggested that abstraction would have been unnecessary.

Most players experienced one or more losses against the computer while playing the fifteen game and the word game. As a result, these players were presented with the abstracted versions of the fifteen game and the word game. All of these players experienced improvement while playing the abstracted versions. All of them tied the computer except for one, who beat the computer.

Each player was briefly interviewed after the experiment. They were invited to comment on the games, the process, and what they had learned. Some of these players commented that they felt the fifteen game and word game might be similar to tick-tack-toe (before seeing the abstracted versions), but were unsure of why or in what way they might be similar. Many of the players knew about magic squares, but did not know about using them with these games. All the players agreed that the abstracted versions of the fifteen game and the word game were easier to play.

The game playing problem in this experiment did not require explicit representation. More complex problems (one is discussed in the following sections) will require detailed representations of both the knowledge domain and individual perspectives.

3. A MORE COMPLEX PROBLEM

At Los Alamos National Laboratory, researchers frequently use complex computer simulations to model experiments and to explore physical relationships. These simulations are used to predict the performance of some physical device. The process of setting up of these computer models can be extremely complicated. Typical input specifications contain hundreds to thousands of specified inputs in addition to specifications of computational representations of geometry (requiring millions to billions of specifications). As a result, code manuals are lengthy, set-up times are long, and once an input specification is developed it is reused with minor perturbations. In addition, the codes are designed with common defaults in place so that the user provides a subset of all of the data that is truly required to specify a particular simulation. Once created, legacy input specifications (decks) may be passed from user to user and can be as critical to the success of modeling efforts as the code itself.

The problem with this situation is that a code user can easily fall into the trap of running codes without even realizing that certain critical input specifications have defaulted to inappropriate values. Less experienced users may even generate false results with no indication that a problem exists.

Training of users in this situation is difficult. The users come from a variety of background disciplines, and the codes typically represent a multi-disciplinary integration. In a typical code, the number of input variables is so large, and the interaction is sufficiently complex that there is a large number of cases to be trained upon. Simply dropping a user into the abyss with code manual in hand is generally certain to overwhelm the user making the learning curve impossibly steep. In the past, mentors have been used to guide the trainee through the process of discovering the proper use of these codes. However, availability of mentor time and skilled mentors is limited. Another approach is desirable.

Ultimately, our goal is to use the notion of perspectives to help train and guide users of these complex integrated packages. The training of each individual combination of input possibilities is hopelessly large and probably not necessary. Instead, the function of the input variables can generally be grouped as those that are relevant to a particular perspective of the problem to be solved. For example, description of geometry, material properties, and initial conditions can be grouped into a "Geometric Perspective." These notions would be easily understood and quickly grasped as a group by, for example, recently graduated engineers. Grouping variables that control physics model selections such as turbulence models, isotopics, transport approximations, or scattering mechanisms into a "Physics perspective" creates a collection of variables that will be nearly intuitive to a physicist. The computational solution of the proposed physical problem always involves mathematical approximation and algorithmic solution. Grouping variables such as convergence tolerances, flags for determining iteration methods, flags for determining mathematical approximations (e.g. element order in a finite element code) into an "Applied mathematics perspective" will collect the description variables that are intuitive to most numerical analysts.

A well trained, competent user should have knowledge of all these perspectives, but typically, fresh users (and experienced users that are in a rut) do not have this knowledge. To help maintain the competence of these code users we propose to archive the history of input decks used by each user, and use pattern matching to determine in which of these broad perspectives the user operates or is operating. For example, if we see frequent changes to a geometry, but no changes in convergence criteria or algorithm selection we may deduce that the user *sees* the code from a "Geometric perspective." If there are rarely errors in the input deck specification we may deduce that the user is competent within this perspective. As a result, the user can be informed of the existence of another perspective. This could take place by changing some of the variables in one of the other perspectives to be required input rather than a default *for this user* thereby requiring this user to broaden their understanding of the simulation process. In

this way, individual users are trained at an appropriate individualized rate to develop a broader understanding of their modeling process.

Clearly, in this more complex problem, representation of the perspective is more difficult. In the example game playing problems of tick-tack-toe and its variants, a simple mapping (essentially an analogy) is sufficient. In this more complex problem detailed interactions between the input variables must be represented. This drives the use of a representation methodology for the complex perspectives.

4. OBJECT-BASED KNOWLEDGE REPRESENTATION

4.1. Introduction

The notion of perspective is supported in an object-based knowledge representation methodology created by the authors. This methodology helps design, create, and use knowledge representations. The created representations are easily decomposed, closely coupled with the actual knowledge, easily maintained, non-ambiguous, structured, and modular.⁵

The methodology contains a set of meta-level constructs. These constructs are used to create a meta-representation of the knowledge to be represented. The meta-representation can be considered an ontology, but smaller than typical ontologies and specifically for the knowledge to be represented. A typical ontology attempts to make available categories to contain any and all knowledge (one size fits all), thus they are large and complex. The point of the meta-representation is not create ontologies, but to create a close coupling between the representation and the knowledge being represented. These meta-representations (or small specific ontologies) are domain specific and not meant to be further used for other domains of knowledge. Each domain of knowledge and its applications should create a new ontology (unless a match occurs, in which reuse is encouraged).

The following sections briefly describe the object-based knowledge representation methodology. Particular attention is paid to the methodology's support for the notion of perspective. More detailed information and examples of the methodology's use can be found in Kelsey et al.^{6,7,5}

4.2. Meta-level constructs

A set of primitive and meta-level constructs is contained within the methodology. These constructs are the basic building blocks for designing a meta-representation for the representation of a specific body of knowledge, such as the knowledge necessary for a particular application. Relationships between the meta-level constructs are also defined. These relationships support how the meta-level constructs fit together and how they are used for creating a meta-representation.

The most basic of the meta-level constructs are the class, domain, and event constructs. Each of these constructs can contain additional constructs. A class can contain attribute, method, and perspective constructs. A domain can contain instance constructs. An event can contain a before state, meth (method), after state, and agent constructs.

A class represents a collection of objects. It can be considered a template for creating an instance or object. The class is the most fundamental of the constructs as it creates what is potentially available for use in the body of knowledge. A class is defined and described by its attributes and methods. An attribute is a characteristic or property of an object of the class. A method is an operation on an object of the class and can demonstrate behavior.

A perspective construct is defined within a class but is not a component for describing a class. A perspective defines a point of view through which an object of the class views other objects of other classes. It can define how those other objects are seen and perceived. A perspective can also be used to further focus and limit knowledge being represented.

A domain construct defines a group of objects that are instantiated from the available defined class. As such, a domain contains instance constructs which are instances (objects) of the classes. An instance represents an actual and unique object as defined by a class. Where a class is a template, an instance is the actual entity. A domain construct with its declared instance constructs represents a body (domain) of knowledge for use in an application.

An event construct defines an actual use of the knowledge represented in the domain construct. This is not to be confused with a triggering mechanism or triggering entity. The construct is a use description or demonstration that occurs between two objects. The before state construct is a list of attributes and their associated values (of the two objects) that change during the event being represented. Non-changing attributes are not listed and the associated

values in the before state are values before the change occurs. The meth construct is the method (operation) causing the change to occur. The after state construct lists the same attributes of the before state but with their values after the change has occurred. The agent construct identifies the object who called the meth (method) to instigate the change.

A number of relationships between the meta-level constructs are built in and supported to create structure in the meta-representation. This allows the structure and relationships in knowledge (being represented) to be better represented. These relationships are inheritance, aggregation, perspective, and method-based. Using inheritance, a class can be defined to be a child of another (parent) class. The child class inherits all the attributes and methods of the parent class.

Using aggregation a class can be defined that is made of components and each component is a class, or rather an object of a class. This is accomplished by allowing an attribute construct of a class to contain an instance construct. In this manner aggregate classes of other classes are created. A class can possess a personalized, limited, and focused view of other classes in a domain. This is the notion of perspective.

There are other relationships that are specific to the knowledge being represented. These are known as method-based relationships. These relationships occur between objects of classes that interact with each other through methods defined in each of the classes. Methods are the operations and behavior specific to objects of a class and this is the reason why these relationships are called method-based and specific to the knowledge being represented.

4.3. Implementation

The meta-level constructs and their use is implemented in the Standard Generalized Markup Language⁸ (SGML). There are a number of reasons for using SGML to implement a knowledge representation scheme and methodology. SGML provides benefits that are not normally available in other representation schemes. SGML is a markup language and there is growing familiarity with markup languages and their use. An example is the hypertext markup language (HTML) which is used for World Wide Web (WWW) documents. A document type definition (DTD) for SGML actually defines HTML and the Web browsers for displaying and reading HTML documents. In a manner of speaking, HTML is a sub-language of SGML.

Another benefit of SGML is its portability. Again, HTML Web documents are a good example. Web documents written in HTML can be displayed and read on many different hardware platforms using many different operating systems and many different interface software packages. Additionally, small and simple post parser routines can be created to translate SGML to other languages and systems as necessary.

A DTD provides a formal definition of the modules (in this case) in the meta-representation and how those modules relate to one another. This defines structure of and between the modules. The DTD is actually a formal grammar that defines the syntax of the meta-representation. Syntax is defined in that the markup language for each module (construct) is defined and the order and relationships between modules is defined.

Equally beneficial and important is that the DTD can be used to enforce the structure and syntax of a representation. Other knowledge representation systems and schemes tend to be permissive in that a construct can be used for more than its original intent. For example, frames can be defined and used in many different ways and that is part of their problem. Using the proposed methodology to represent knowledge will not allow such permissiveness because the knowledge can be parsed and checked against the DTD.

4.4. Perspective

In an object-based knowledge representation an entity can possess a personalized, limited, and focused view of other entities in a domain. This perspective (or view) can change as necessary. In the game playing experiment, each player was an entity in the knowledge domain of interest. Each player possessed an individual perspective representing how to play each game and game strategies to apply. In this case, the perspective construct embodied a piece of knowledge for game playing. As a player's perspective was adapted and evolved, new knowledge became available to the player.

5. CONCLUSIONS

The game playing problem was chosen because of its simplicity and because it demonstrated well the advantages of a change in perspective. The experiment results and interviews with players confirmed that adapting and changing a perspective (in the example problem) can lead to better understanding and reasoning about the problem. Also, the notion of perspective and its representation can be used to guide and train users in a "stepping stone" manner to higher levels of expertise and knowledge.

Although the game playing problem did little to utilize the extent of the object-based knowledge representation methodology, the methodology is expected to be useful and necessary for more complex problem domains. This is becoming apparent in initial decomposition of the simulation setup problem.

REFERENCES

1. J. R. Anderson, *Cognitive Psychology and Its Implications*, W.H. Freeman and Company, New York, NY, fourth ed., 1995.
2. N. Maier, "Reasoning in Humans: II. the solution of a problem and its appearance in consciousness," *Journal of Comparative Psychology* 12, pp. 181-194, 1931.
3. M. Gardner, *Aha!*, Scientific American, Inc. / W.H. Freeman and Company, New York, NY, 1978.
4. W. S. Andrews, *Magic Squares and Cubes*, Dover, 1960.
5. R. L. Kelsey, *Object-Based Knowledge Representation Implemented in SGML*. PhD thesis, New Mexico State University, Las Cruces, NM, July 1998.
6. R. L. Kelsey, R. B. Webster, and R. T. Hartley, "Using multiple perspectives to suppress information and complexity," in *Digitization of the Battlespace III*, vol. 3393, pp. 72-85, Society of Photo-Optical Instrumentation Engineers, Society of Photo-Optical Instrumentation Engineers, (Bellingham, WA), 1998.
7. R. L. Kelsey, R. T. Hartley, and R. B. Webster, "An object-based methodology for knowledge representation in SGML," in *Proceedings of the Ninth IEEE International Conference on Tools With Artificial Intelligence*, pp. 304-311, IEEE Computer Society, (Los Alamitos, CA), 1997.
8. International Organization for Standardization, Geneva, *ISO 8879:1986 Information processing - Text and office systems - Standard Generalized Markup Language (SGML)*, October 1986.